NPS54-81-014

NAVAL POSTGRADUATE SCHOOL

Monterey, California

STOCKPOINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT (SPLICE) NETWORKING STUDY

Norman F. Schneidewind

October 1981

Final Report for Period March 1981 - October 1981

Approved for public release; distribution unlimited

Prepared for:
Fleet Material Support Office
Mechanicsburg, Pennsylvania

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral J. J. Ekelund
Superintendent

David R. Schrady
Acting Provost

The work reported herein was supported by the Fleet Material Support Office.

This report was prepared by:

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1 REPORT NUMBER: NPS-54-81-014	2 GOVT ACCESSION NO.
3 RECIPIENT'S CATALOG NUMBER	
4 TITLE (and Subtitle): STOCKPOINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT (SPLICE) NETWORKING STUDY	5 TYPE OF REPORT & PERIOD COVERED: Final Report 1 March 81 to 31 Oct 81
	6 PERFORMING ORG. REPORT NUMBER
7 AUTHOR(s): Norman F. Schneidewind	8 CONTRACT OR GRANT NUMBER(s)
9 PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School Monterey, California 93940	10 PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: N0003671WRN3880
11 CONTROLLING OFFICE NAME AND ADDRESS: Fleet Material Support Office Mechanicsburg, Pennsylvania	12 REPORT DATE: 31 October 1981
	13 NUMBER OF PAGES: 73
14 MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15 SECURITY CLASS. (of this report): Unclassified
	15a DECLASSIFICATION/DOWNGRADING SCHEDULE

16 DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17 DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18 SUPPLEMENTARY NOTES

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Networks
SPLICE
AUTODIN II
Protocol
Interface

20 ABSTRACT (Continue on reverse side if necessary and identify by block number)
The Stockpoint Logistics Integrated Communications Environment (SPLICE) Network is to provide a networking capability for Navy stockpoints and inventory control points. One alternative for providing this capability is AUTODIN II. If it is to be utilized, an interface between SPLICE and AUTODIN II is necessary. This involves consideration of hardware, software and protocols. This report provides an overview of the various AUTODIN II protocols which are necessary for the SPLICE-AUTODIN II connection alternatives and a recommended connection; and an evaluation of a Navy-designed AUTODIN II Interface and a proposal for a modified interface.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 68 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

SUMMARY

This is the final report on networking options--specifically on AUTODIN II interconnect options--of the SPLICE Networking Study. The following project items are covered in this report:

- Protocols
- Virtual and physical links
- Data flows (flow control)
- Log-on procedures (at the level of open and close processing and connection establishment)
- Economic factors
- Network topology and connectivity
- AUTODIN II interface options

The report is not organized by the above topics. For continuity of flow and reader understanding, the report is organized by the following topics:

- Background
- Overview of AUTODIN II System (with orientation to SPLICE)
- Terminal-To-Host Protocol
- Transmission Control Protocol
- Segment Interface Protocol
- Advanced Data Communiction Control Procedure
- Recommended SPLICE-AUTODIN II Connection
- Navy AUTODIN II Interface Design
- Navy AUTODIN II Interface Design Recommendations

The recommendations of the report are as follows:

- The Solicitation Document (SD) should include a mandatory requirement which calls for the minicomputer front-end processor to

provide a single AUTODIN II interface at each host site which will allow communication between terminals, between hosts, and between terminals and hosts. Additionally, any protocol software to be provided must be coded in a single HOL, acceptable to the Navy.

- The SPLICE-AUTODIN II interface referenced in the above recommendation should be in accordance with the specifications of Figure 6 of this report and the accompanying narrative. Furthermore, this interface should be acquired via the SPLICE procurement process.

- The Interdata 7/32, as described in references 16 and 17, should not be used as the hardware for SPLICE.

- However, much of the software design, as documented in reference 17, should be used as the baseline for the SPLICE software design, incorporating corrections to the deficiencies noted in Section VIII and changes recommended in Section IX.

TABLE OF CONTENTS

LIST OF FIGURES

I. BACKGROUND

Early in this project it became clear that the connection and interaction of SPLICE with AUTODIN II would be a major determinant to the success of SPLICE for the following reasons:

- The performance characteristics of AUTODIN II will exert a profound influence on the speed, accuracy, security and reliability with which messages can be delivered in the SPLICE network. Additionally, the need for interactive use and file transfer, both terminal-to-host and host-to-host over widely separated node locations, requires a network with satisfactory response and transit times.
- There are non-trivial considerations regarding the alternatives for connecting and interfacing SPLICE to AUTODIN II. Each of the alternatives has advantages and disadvantages with respect to ease of installation, network complexity, maintainability and life-cycle support, and cost.
- There are important ramifications concerning the nature of the connection of SPLICE to AUTODIN II with respect to the Request for Proposal (RFP) which is to be issued for the SPLICE system. Specifically, the hardware and software used in the SPLICE mini-computer front-end processor at each node must be compatible with the AUTODIN II interface hardware and software. The various alternatives for achieving compatibility could have a significant effect on acquisition strategy.
- Commercial data communication networks could be considered as an alternative to AUTODIN II provided it can be demonstrated that

such an arrangement would provide the degree of security, survivability and continuity of service that AUTODIN II is projected to provide.

For these reasons, AUTODIN II should not be considered incidental to the operation of SPLICE, but, rather, as an important integrated element of the SPLICE concept. Given this perspective, the emphasis in this report is on the evaluation of AUTODIN II connection alternatives and the design of the interface between SPLICE and AUTODIN II.

For readers who are unfamiliar with AUTODIN II, several descriptive sections have been provided which give an overview of the network--it's hardware and software components--followed by a description of each of the non-user specific protocol layers of AUTODIN II. This treatment is not intended to be exhaustive or to cover every conceivable communication situation in AUTODIN II. Only those points which are necessary for understanding the overall network operation and which are most relevant to interface design are covered. The very detailed referenced documents provide the definitive specifications for interface design. An objective of this presentation is to provide a clear and succinct description of a complex technical subject. The next sections provide a discussion of connection alternatives. The report closes with a description of a recommended interface.

II. OVERVIEW OF AUTODIN II SYSTEM

The AUTODIN II System can most conveniently be described in terms of the major sections of access area and backbone. A further breakdown can be made to the component level within each of these major sections [8, 9, 10, 11, 12, 13]. Access areas and backbone are shown in Figures 1 and 2. In the diagrams which follow both the possible and the recommended connections of SPLICE to AUTODIN II are delineated. The former represent the conventional interfaces which are provided by AUTODIN II. The latter represent the interfaces and connection methods which are felt to be most advantageous for SPLICE. Before discussing these alternatives, the various hardware units and protocol units which are used in AUTODIN II are defined below.

PROTOCOL	DEFINITION
Terminal Handler (TH)	Terminal dependent handler which interfaces terminals to THP. May reside in TAC or FEP.
Host Specific Interface (HSI)	Host dependent handler which interfaces a host to THP. May reside in CCU or FEP. Counterpart of TH for hosts.
Terminal-To-Host Protocol (THP)	Initiates open and close processing, communication processing of user data and Network Virtual Terminal services. Interfaces TH or HSI to TCP. May reside in TAC, CCU or FEP.

Transmission Control Protocol (TCP)	Establishes, maintains, and closes virtual connections and provides flow control. Interfaces THP to SIP. May reside in TAC, CCU or FEP.
Segment Interface Protocol (SIP)	Provides procedures and rules for exchanging data between TCP and the backbone and segments user data. May reside in TAC, CCU, or FEP.
Advanced Data Communications Control Procedure (ADCCP)	Responsible for physical transport of data on access channel and backbone (between CCU or FEP and LTU, between Mode VI terminal and LTU and between LTU's on backbone). Synonomous with Mode VI. Implemented in the LCM's and LTU's.

HARDWARE COMPONENTS	DEFINITION
Terminal Access Controller (TAC)	Conventional terminal interface to AUTODIN II. Located in PSN. Contains TH, THP, TCP and SIP.
Channel Control Unit (CCU)	Conventional host interface to AUTODIN II. Located at user site. Contains HSI, THP, TCP and SIP.

Front End Processor	User supplied communications processor. Interfaces host and terminals to AUTODIN II. Could contain TH, THP, TCP, and SIP.
Switch Control Module (SCM)	Packet switching component. Makes packets out of segments. Located in the PSN.
Line Termination Unit (LTU)	Terminates lines, converts data from serial to parallel and vice versa, and provides error detection and correction. Located at PSN and user site.
Line Control Module (LCM)	Multiplexes and demultiplexes data between LTU and TAC, CCU, or SCM. Monitors line status. Located at PSN and user site.
Parallel Communication Link (PCL)	Parallel, high bandwidth channel between various components of a PSN. (Similar to the "UNIBUS.")
Switching Node	Consists of one or more TAC, SCM, LTU, LCM and PCL. Located at AUTODIN Switch Center (e.g., McClellan).

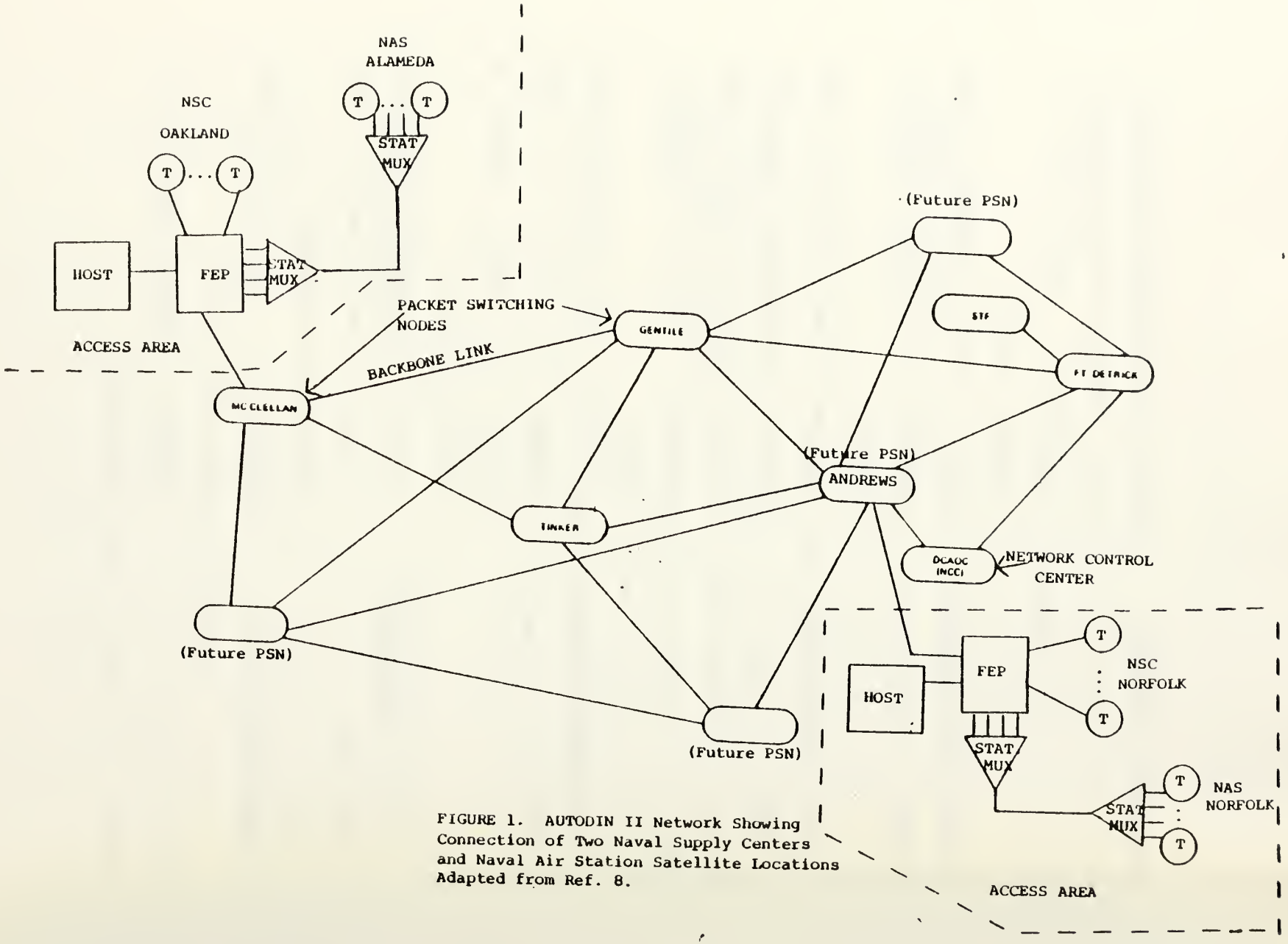


FIGURE 1. AUTODIN II Network Showing Connection of Two Naval Supply Centers and Naval Air Station Satellite Locations Adapted from Ref. 8.

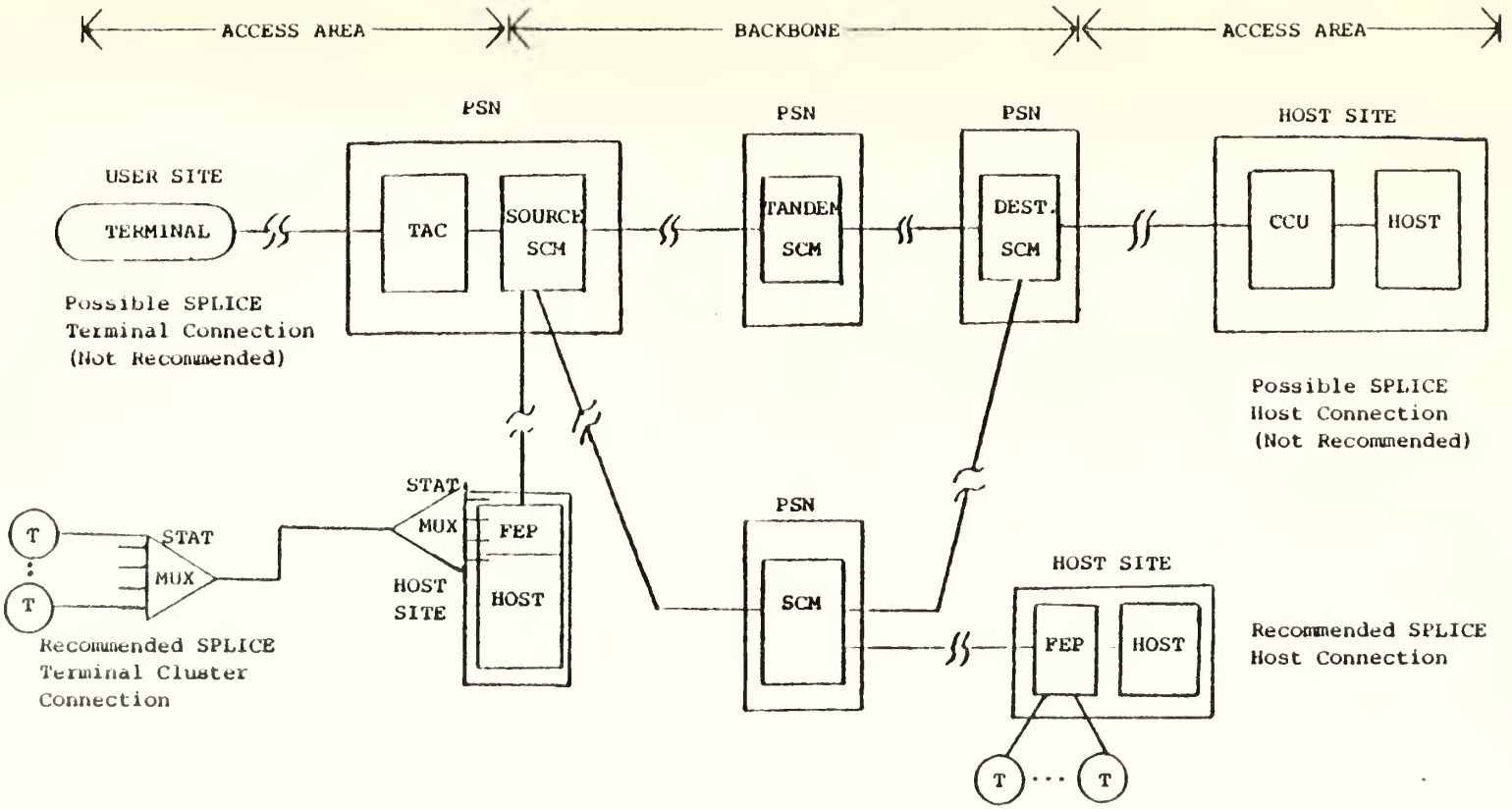


FIGURE 2. Diagram of Possible and Recommended Terminal and Host Connections using AUTODIN II Adapted from Ref. 11.

AUTODIN II divides user access into two categories, terminal access and host access, with an interface unit for each -- Terminal Access Unit (TAC), located in the Packet Switching Node (PSN) and Channel Control Unit (CCU), located at the user site, respectively. Examples of these connections are shown in Figure 2. This arrangement provides a convenient and possibly economical way of connecting to AUTODIN II for organizations which have many geographically dispersed terminals and few hosts. An organization in this situation could run lines from its terminals or intervening multiplexer or concentrator to the nearest PSN, where the interfacing equipment -- Line Termination Unit (LTU) and Line Control Module (LCM) -- are located. This organization, with relatively few hosts, could obtain Multiple Channel Control Units (MCCU) -- a PDP11/34 Channel Control Unit which supports many virtual (Host and Terminal) connections -- from the vendor (Digital Equipment Corporation) or (as part of the CSIF) from the Western Union Telegraph Company, the prime AUTODIN II contractor. However, many organizations, including SPLICE, have many terminals co-located with the host (e.g., Naval Supply Centers); and in addition, have many terminals located at satellite facilities (e.g., Naval Air stations) which are a relatively short distance from a host location. Furthermore, terminals which are co-located with a host and satellite terminals will access the host in addition to other hosts and terminals in the SPLICE network via AUTODIN II. Under these operational requirements, the connection arrangement shown in Figure 1 and the recommended terminal and host connections shown in Figure 2 are more appropriate for SPLICE than are the conventional AUTODIN II terminal and host connections, also shown in Figure 2. The recommended SPLICE connection involves the use of a Front End Processor

(FEP) as shown in Figures 1 and 2; this would be the SPLICE standard minicomputer. This computer will be obtained as part of the SPLICE acquisition. It is to provide communication processing as well as other services. With appropriate storage capacity, I/O facilities and processing speed, it could provide the interface to AUTODIN II as well. As shown in Figures 1 and 2, satellite terminals would be multiplexed, using a statistical multiplexer (or concentrator), and connect to the FEP at the host site. Also, as shown in Figure 2, this alternative does not require the use of the TAC at the PSN; the line from the FEP is run directly to the SCM at the PSN. This means that separate connections for terminals and host to AUTODIN II are not necessary nor are CCU's necessary, as with the conventional AUTODIN II connection philosophy. Instead, the FEP which has to be interfaced with SPLICE terminals and acquired anyway, serves as the single interface, thus reducing the number of connection points, number of lines and line mileage, and the complexity of the SPLICE network. Of course, the AUTODIN protocols -- THP, TCP, SIP and ADCCP -- must be provided in the FEP and associated line interfacing equipment, along with the SPLICE-specific terminal and host protocols. More will be said about this later. The recommended connection could be called a software solution as contrasted with the hardware solution of the conventional AUTODIN II.

In order to set the stage for discussing the implementation of protocols under the recommended connection, the various AUTODIN II protocols are described in the following sections. As an aid to the reader for understanding the protocol descriptions, Figures 3 and 4 are provided; these show the conventional AUTODIN II connections and protocols for terminal and host access, respectively.

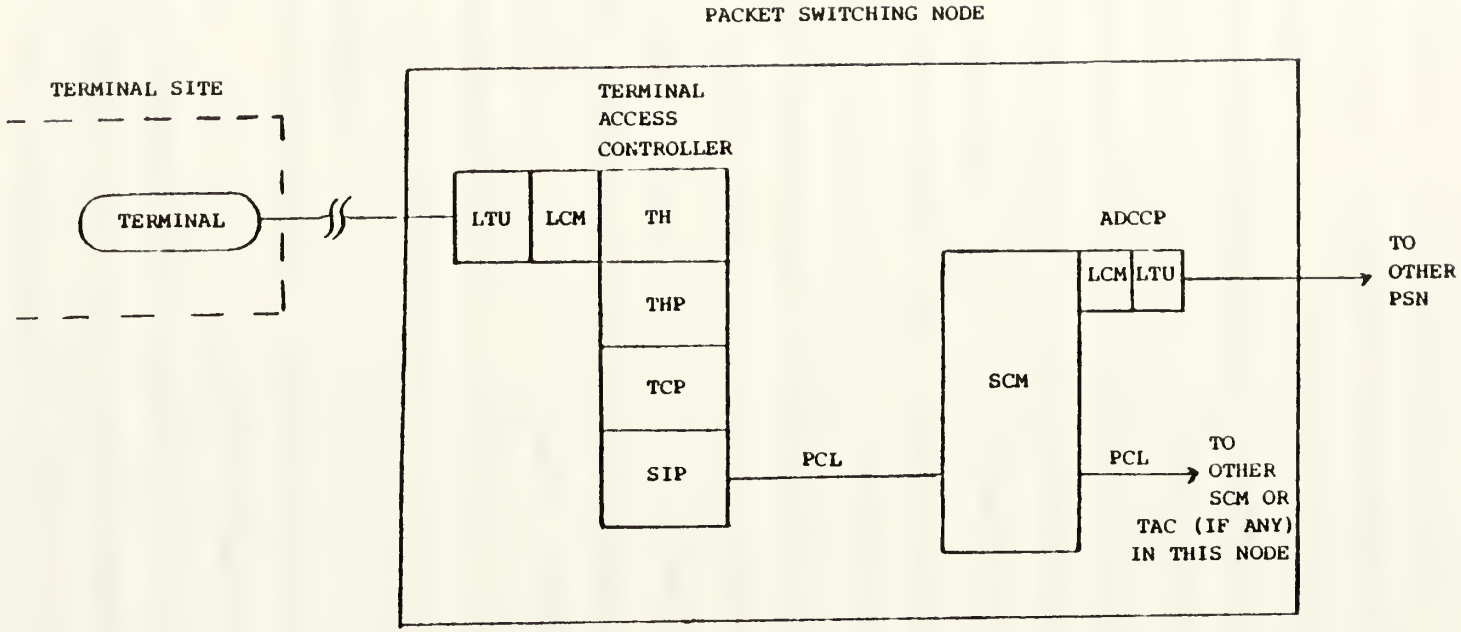


FIGURE 3. Possible Terminal Access to AUTODIN II (Not Recommended)
Source: Ref. 11.

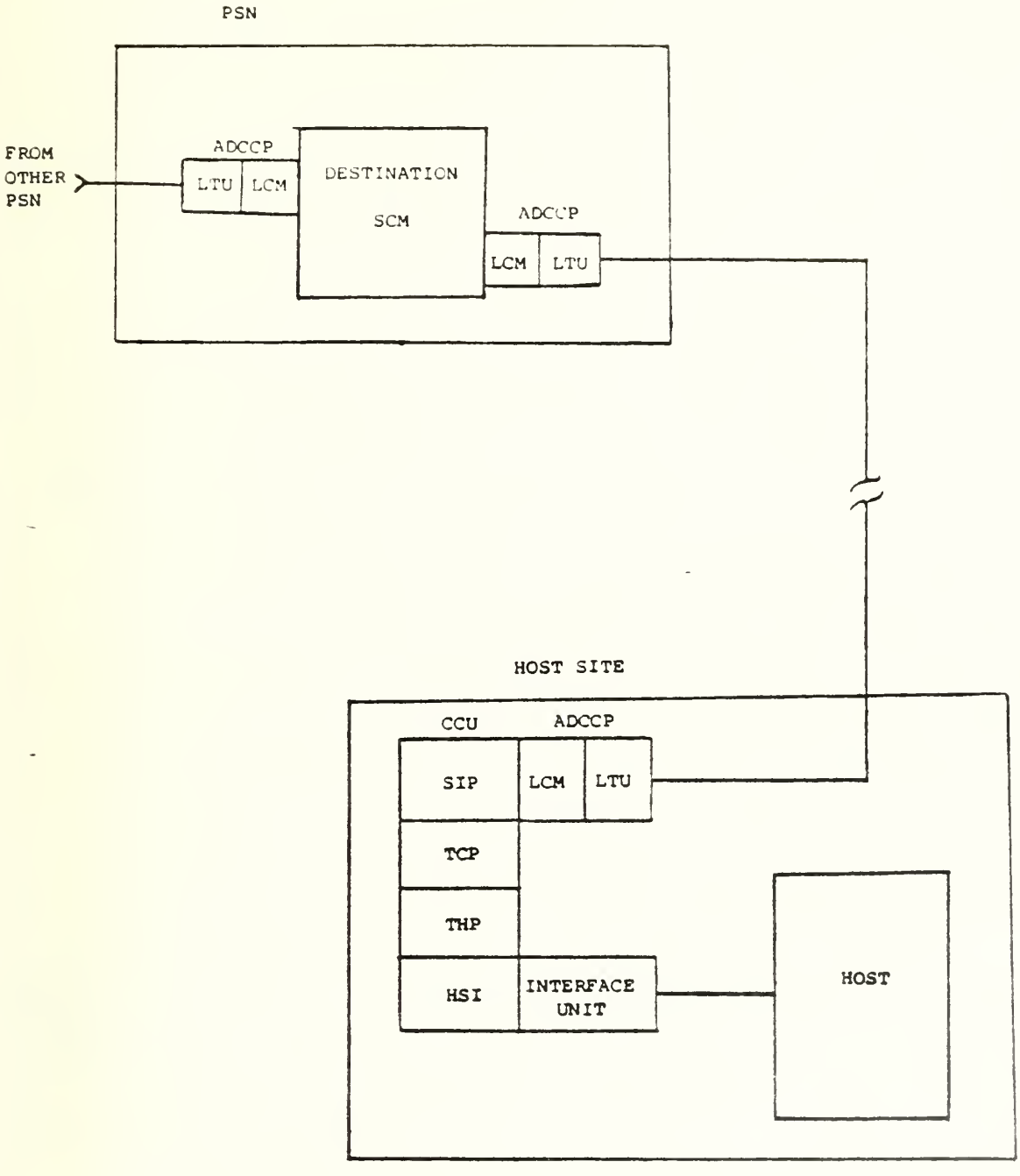


FIGURE 4. Possible Host Access to AUTODIN II
Via a Channel Control Unit
(Not Recommended)
Source: Ref. 11.

III. TERMINAL - TO - HOST PROTOCOL

The Terminal-To-Host Protocol (THP) is the second layer (from the top) protocol of AUTODIN II [1]. THP Performs the following services:

- Provides communications processing of user data going to and coming from the network, where "user" may be a human operator at a terminal or a computer process.
- Requests user-oriented services from the third layer protocol, Transmission Control Program (TCP).
- Converts from user format to Network Virtual Terminal (NVT) format for data going from user to network, and from NVT format to user format for data going from network to user.
- Provides binary mode for data to or from network.
- Initiates the opening of a connection.
- Initiates the closing of a connection.
- Performs error detection of invalid or illegal commands.

Network Virtual Terminal

Fundamental to an understanding of the operation of the THP is the concept of the Network Virtual Terminal (NVT). The NVT idea was borrowed from ARPANET for use in AUTODIN II. The NVT defines an AUTODIN II internal standard bidirectional character oriented device. The NVT is a hypothetical terminal, with specified characteristics onto which source user data is mapped for transmission on the network [2]. A second mapping takes place from NVT characteristics to destination user data. Thus, the NVT serves as a translator between terminals with different characteristics. The "terminal" can be either a user terminal or a computer. In general, the mappings are many to one from source

terminals to NVT and one to many from NVT to destination terminals, where "many terminals" refers to both quantity and variety.

In AUTODIN II, NVT is the normal or default mode of operation. However, this may be changed by means of a process called option negotiation, which is described below. This is another concept adapted from ARPANET. The source THP formats data destined for the network as NVT printer format, according to the source user's line width and page size. The NVT printer can represent 95 ASCII graphics and 33 ASCII control codes. Of the 33 ASCII control codes the following eight cause the indicated operations to take place on the NVT printer:

- Carriage Return (CR): moves print head to left margin of current line.
- Linefeed (LF): moves print head to next print line.
- Horizontal Tab (HT): moves print head to next tab stop.
- Vertical Tab (VT): moves print head to next vertical tab stop.
- Backspace (BS): moves print head one character position to the left.
- Bell (BEL): produces an audible signal.
- Null (NUL): provides timing delay, but produces no printer function.

Another six codes (data transmission control characters) which do not involve print operations but which are part of the NVT model are the following:

- Send Now (SN): causes accumulated user data to be sent to the network.
- Are-You-There? (AYT?): causes a character to be sent to a remote user which requests the user's status.

- Erase Line (EL) or Erase Character (EC): sent to remote user to perform indicated function.
- XASCII Shift-Out (SO) and Shift-In (SI): causes data between SO and SI to be sent in XASCII code.
- Go-Ahead (GA): causes GA character to be sent to a remote user indicating sender is ready to receive data.
- Interrupt Function Characters (IF): causes various interrupt functions to be performed by local or remote THP/TCP.

Record Modes

User data is transmitted in two modes -- record mode and stream mode. In record mode both user text and THP control are packaged in THP records. In stream mode only THP control is packaged in THP records; user text is sent anywhere in the stream without record delimiters. User data is further packaged by combining one or more THP records into a THP letter. The letters are sent by THP to TCP for transmission on the network. A letter is released by THP to the network when the THP detects one of the following conditions:

1. Receipt of a unit of user text (one or more characters).
2. Receipt of an end-of-line character.
3. Receipt of a specified number of characters, up to a maximum of 584.
4. Receipt of a Send Now (SN) character.
5. Receipt of maximum segment size of 584 octets (8 bit units of data) or user's maximum buffer size.
6. Receipt of an Are-You-There? (AYT?) character.

7. Receipt of an RCTE command (a remote echo control command). For this release to occur, the RCTE option must have been negotiated.

8. Receipt of an Interrupt Function (IF) character.

9. Receipt of a Go-Ahead (GA) character. For this release to occur, the Go-Ahead option must have been negotiated.

Of the above methods, any one of the first four (1-4) can be selected or changed by the user via the Packet Release Command. This will be the release mechanism used unless one of the last five conditions (5-9) occurs first.

Scanning User-To-Network Data

There are four methods used by the THP to scan and recognize user data. These methods differ by the characters which are recognized and the processing which ensues. The four methods are the following:

- NVT Mode: This is the network default mode. The control characters CR, FF, LF, HT, VT, BS, BEL and NUL and the data transmission control characters SN, AYT?, EL, EC, SO, SI, GA and IF are recognized and acted upon. All characters are transmitted as 7 bit ASCII (the parity bit is stripped from each character). If control characters are preceded by a prefix character, (PC), the control characters are passed as data or considered an invalid sequence, depending upon circumstances. If not preceded by a prefix character, the action called for by the control character is taken. The typical prefix character is the "at" sign (@), but almost any special character may be set.

- Transparent Mode: This mode is started and stopped via the Transparent Command. The purpose of this mode is to allow sending of data control characters as data without requiring each control character or series to be preceded by a Prefix Character, as in the NVT Mode. For control characters to be recognized while in Transparent Mode, they must be preceded by a Prefix Character. As in NVT Mode, the parity bit is stripped from characters prior to transmission.

- Binary Mode: The purpose of this mode is to transmit data as is, without causing the recognition and processing of control characters. However, control characters will be recognized if preceded by a Prefix Character. Unlike the NVT and Transparent Modes, the parity bit is not stripped; eight bit characters are transmitted. Binary Mode is automatically entered when opening a connection if the local and remote user characteristics match. The Binary Mode can be requested by the user issuing the Binary Mode Option Command.

- XASCII Mode: In this mode no control characters are recognized except Shift-In (SI). The purpose of the mode is to allow the user to send data as pure XASCII records, with no recognition of control characters, and no stripping of parity bits. The XASCII Mode is negotiated via the XASCII Option Command. If agreement is reached, the mode is entered when SO (Shift-Out) is recognized in the user data stream. The mode is exited when the SI character is detected.

Processing Network-To-User Data

If data coming from the network is packaged in Stream Mode (see previous Record Modes section), every character must be examined for a possible control character. If the data is packaged in Record Mode, all

data is formatted as records and it is unnecessary to search for control characters.

Data is released from the network to the user by the THP according to one of the methods listed below, which is established at system generation time and is not changeable:

- Receipt of a specified number of characters (from 1 to 584).
- Receipt of a THP letter.
- Receipt of an End-Of-Line character.

Option Negotiation

Option negotiation (another concept borrowed from ARPANET), allows the local THP to negotiate with the remote THP for changes to the normal NVT mode of operation. In order for the option to take effect, both parties must agree to the option. The local THP negotiates on the basis of the user's profile or option commands entered by the user. Option negotiations will occur automatically during open processing; this is known as the Characteristics Option. This option consists of two parts -- compatibility check and charcteristics check. The purpose of the former is to determine whether the local and remote users are compatible according to the specifications of the Cross-Connection Matrix (e.g., TTY Type Port allowed to connect to a programmable CRT Type Port but not to a Magtape/Card Type Port). If this check fails, the connection will be closed immediately. The characteristics check determines whether the two users have matching characteristics. If this is the case, transmission will occur in Binary Mode rather than NVT Mode, since the former incurs less overhead.

The second situation in which negotiation can occur is when the user enters an option command. The following options are available:

- Binary Mode (specify this mode).
- RCTE (use the specified echo characteristic).
- Go-Ahead (recognize this symbol).
- XASCII Mode (specify this mode).
- Line Width (specify width).
- Page Size (specify size).
- Horizontal Tab Stops (specify location of stops).
- Vertical Tab Stops (specify location of stops).
- Carriage Return Disposition (transmit, discard or add delay).
- Linefeed Disposition (transmit, discard, simulate character, or add delay).
- Formfeed Disposition (transmit, discard, simulate character, replace with new line or add delay).
- Horizontal Tab Disposition (transmit, discard, simulate character or add delay).
- Vertical Tab Disposition (transmit, discard, simulate character, replace with new line, or add delay).

A THP at each end of the virtual connection is involved in option negotiation. Each THP has a send data path and a receive data path. Only one of these data paths is negotiated per option request. For example, the local THP could request to start an option on its send data path and agree to do the required processing. In response, the remote THP could agree to start an option on its receive data path and agree that the data sender THP do the processing. On the other hand, the remote THP could refuse the request.

IV. TRANSMISSION CONTROL PROTOCOL

The Transmission Control Protocol (TCP) is the third layer (from the top) protocol of AUTODIN II [3]. TCP is concerned with connection processing and provides the following services:

- Performs the mechanics of establishing, maintaining and terminating a virtual connection, where "virtual" means a logical connection between two users as opposed to a physical connection.
- Provides the mechanism by which THP can communicate with the network. The THP-TCP relationship is an example of communication between different layers in the same node as opposed to peer protocol communication involving the same layer in different nodes (e.g., THP-THP).
- Provides flow control so that users at both ends of virtual connection appear to be communicating via a dedicated physical circuit.

Sequence Control

AUTODIN II follows the ARPANET practice of sending the data units which constitute a message in parallel over many routes from local user to remote user. Since sequence is not maintained during transmission, data units must be reassembled at the destination. For this purpose, a sequence number is associated with each segment of the data unit of the TCP's. Other uses of sequence numbers are to ensure the validity of a received segment on a given virtual conenction, provide a means for the receiving TCP to acknowledge a segment to a sending TCP and for detecting duplicate segments.

In order to provide flow control (avoid excessive congestion), TCP utilizes a window. The window is defined as the maximum number of octets awaiting acknowledgement by the source TCP from the destination TCP [2]. If n is the number of octets transmitted by the TCP, ℓ the number of octets for which acknowledgement has been received by the source TCP from the destination TCP, the window w is defined by $n - \ell \leq w$. The number of octets transmitted n must be controlled such that $n - \ell$ never exceeds the pre-selected value of w. This flow control procedure is mechanized by means of the sequence numbers referred to earlier.

Virtual Connection Addressing and Sockets

The TCP uses a technique for virtual circuits and connection addressing known as sockets (ala ARPANET). A socket is an addresss which consists of the concatenation of network and port identifiers. The socket defines one end of the virtual connection. A pair of sockets--the source and destination-- completely defines the virtual connection. When a process makes a request to open a conection, both source and destination sockets must be specified.

Since the unit (Channel Control Unit or Front End Processor) which interfaces a host to the network is represented by a single address, the host port ID, corresponding to a particular user process, must be included in the socket address in order to uniquely identify the user process. This problem does not exist for terminals connected to a Terminal Address Controller, since each terminal is individually addressed.

Flow Control

Congestion alleviation takes place at various points in AUTODIN II and at various protocol levels. Flow control is invoked by using the window previously described and acknowledgement procedures. The various points in the network where flow control occurs and the protocols which are involved are summarized below.

- User - To - Network Flow:

  -- Between source HSI (Host Specific Interface) and source THP.

  The source HSI has up to a maximum number of events (meaning control segments in AUTODIN II -- an unfortunate choice of terminology) outstanding (unacknowledged) from the source THP. When the maximum is reached, HSI will no longer accept data from the host. There is no flow control if HSI is asynchronous (Mode IIA) unless an Interface Control Unit is installed.

  -- Between source THP and source TCP.

  The source THP utilizes the THP-TCP send window to regulate segment flow. When the number of unacknowledged events from the source TCP reaches the window size, the THP stops accepting segments from the source HSI.

  -- Between source TCP and destination TCP.

  The source TCP uses the TCP-TCP window as its flow control. This is the maximum number of outstanding octets. The window is changeable by the destination TCP and is transmitted to the source TCP by the destination TCP in every segment. The source TCP can send this many octets before waiting for an acknowledgement from the destination TCP's will

Flow Control

Congestion alleviation takes place at various points in AUTODIN II and at various protocol levels. Flow control is invoked by using the window previously described and acknowledgement procedures. The various points in the network where flow control occurs and the protocols which are involved are summarized below.

- User - To - Network Flow:
  - -- Between source HSI (Host Specific Interface) and source THP.

    The source HSI has up to a maximum number of events (meaning control segments in AUTODIN II--an unfortunate choice of terminology) outstanding (unacknowledged) from the source THP. When the maximum is reached, HSI will no longer accept data from the host. There is no flow control if HSI is asynchronous (Mode IIA) unless an Interface Control Unit is installed.
  - -- Between source THP and source TCP.

    The source THP utilizes the THP-TCP send window to regulate segment flow. When the number of unacknowledged events from the source TCP reaches the window size, the THP stops accepting segments from the source HSI.
  - -- Between source TCP and destination TCP.

    The source TCP uses the TCP-TCP window as its flow control. This is the maximum number of outstanding octets. The window is changeable by the destination TCP and is transmitted to the source TCP by the destination TCP in every segment. The source TCP can send this many octets before waiting for an acknowledgement from the destination TCP. When the number of unacknowledged octets equals the window,

flow between TCP's will cease and the source TCP will be unable to acknowledge the source THP, thus sutting off the THP to TCP flow at the source end.

- Network - To - User Flow:

  -- Between destination TCP and destination THP.

  A maximum value is established for the number of outstanding events between the destination THP and destination TCP. When the number of unacknowledged events from the destination THP to the destination TCP reaches the maximum value, the THP stops sending events to the TCP. This, in turn, causes the traffic from the source to slow down.

  -- Between destination THP and destination HSI.

  A maximum value is established for the number of outstanding events between the destination THP and destination HSI. When the number of unacknowledged events from the destination HSI to the destination THP reaches the maximum value, the HSI stops sending events to the THP. This causes the destination TCP to slow the flow of acknowledgements to the source TCP which, in turn, slows the source TCP transmission rate into the network. This control is not available if asynchronous (Mode IIA) communication is used.

  -- Between destination HSI and the destination host.

  The speed of the destination HSI will be governed by the speed of the output channel and characteristics of the link protocol to the destination host. The rate at which the destination HSI sends events to the destination THP is regulated by the rate of acknowledgement of data sent by the

destination HSI to the destination host. This control is not available if asynchronous (Mode IIA) communiction is used.

Data Formats

Data must be put in correct format for transmission on the network. This format consists of the unit of user data -- a THP letter, a TCP header added to the letter by the source TCP, and a binary segment leader (BSL) added by the Segment Interface Protocol (the fourth layer protocol). The entire package is called a TCP segment or T-segment. After the THP requests of TCP that a letter be sent to the destination user, the TCP formats the segment for transmission on the network.

Acknowledgements

TCP is the recipient of two types of acknowledgements. One is the acknowledgement received from the SIP, indicating that the SCM (Switch Control Module) in the PSN (Packet Switch Node) has acknowledged the segment at the link level. If an error occurs, the nature of the error will be indicated in the acknowledgement. If a link error occurs or the destination host is down, the TCP will attempt retransmission a fixed number of times. Other errors which may occcur are invalid security, precedence or destination address. These errors will cause immediate connection closure.

The second type of acknowledgement is sent by the destination TCP for the number of octets delivered to the destination THP. If acknowledgement is not received within a predetermined time, TCP will retransmit the segment. If a fixed number of retries fail, the connection is closed. The acknowledgement is checked for correct sequence numbers,

which must be between the last acknowledged sequence number and the next sequence number to be sent. If the sequence number is valid, TCP will remove the corresponding octets from the retransmission queue (the queue containing unacknowledged transmitted octets).

Processing Data From the Network

The destination TCP must match data from the network with the correct user. In the case of a terminal user, there is a unique subscriber address; matching user with message does not present a problem. In the case of a host user, a unique subscriber addresss does not exist -- only the host addresss (which is common to a number of users) and the subscriber port ID. The destination subscriber address and port ID specified by the source user during open processing and the source user's subscriber address and port ID constitute a socket pair. The TCP matches the socket pair in the received segment with the socket pair associated with the host user in order to deliver the segment to the THP.

As mentioned previously, the destination TCP must reassemble segments because segments do not necessarily arrive in order. Duplicate and out of range segments must be discarded in this process. To perform this function the destination TCP determines whether the beginning sequence number in the T-segment header and the ending sequence number (determined from length of text) of the received segment are within the range of the receive window. Any out of range octets (segment) are (is) discarded. The acknowledgement sent from the destination TCP to the source TCP contains the sequence number of the next octet expected by the destination TCP.

Close Connection Processing

The TCP is actively involved in closing a connection to a user. Typically, closing a connection will result from a user issuing a CLOSE or ABORT Command to the THP and the THP, in turn, issuing a close request to its associated TCP. However, a close can occur at the initiative of the TCP as a result of a protocol error or by the TCP preempting an existing connection in favor of a higher priority one. Closing can be gradual, as the result of a user CLOSE command, or immediate, as a result of a user ABORT command or protocol error. In a gradual close, the source TCP notifies the destination TCP that close processing is beginning. This will cause the destination TCP to send the remaining data to the destination THP. Once all the octets that have been previously sent to the destination THP have been acknowledged, the source TCP will commence the FIN sequence. This consists of a closing handshake, similar to the connection handshake described previously; source and destination TCP's exchange FIN segments and acknowledgements. One purpose of the FIN sequence is to provide undelivered status information to the user (via the THP) of data that had been entered by the user but had not been segmented (entered into the network) at the time of the close.

In an immediate close, a FIN (flush) sequence begins. This causes data waiting to be transmitted (already segmented) to be flushed; no additional data is sent or received. As before, the FIN sequence provides undelivered status information to the THP.

An existing connection may be preempted by a higher priority (precedence) user. The important conditions which must be satisfied in order to preempt a connection are the following:

- Preempting user must have the security, precedence and transmission control code appropriate for the connection being preempted.
- Preempting user address and port ID must be different than that of the source user address and port ID of the connection being preempted.
- Connection must be eligible for preemption.
- Precedence of the preempting user must be higher than that of the user being preempted.

If a decision is made to preempt, the source TCP sends a "connection preempted" message to the destination TCP and the TCP's engage in close processing for the preempted connection and open connection handshake processing for the new connection, as previously described.

V. SEGMENT INTERFACE PROTOCOL

The Segment Interface Protocol is the fourth layer (from the top) of AUTODIN II [5]/ SIP provides procedures and rules for exchanging data and control information with the backbone (Packet Switch Nodes and associated links) of the network. SIP accepts User Data from the TCP, segments it, attaches the Binary Segment leader (BSL) and passes the segment to a controller for transfer on the SIP-SCM (Switch Control Module) access line or to a TAC co-located at the PSN if so destined. Unlike TCP to TCP communication, control functions are not piggybacked onto data segments. When SIP receives data from the backbone, it detaches the BSL and passes the segment to the user with source and destination user addresses. Once a segment is transferred to the backbone, the SCM divides the segments into packets for transmission on the backbone's data links. SCM attaches source and destination SCM addresses. Packets are reconverted to a segment by the destination SCM. In order to control the flow of data between SIP and SCM, the SCM periodically allocates a window to the SIP. The window is passed from SCM to SIP via the window field of the BSL and replaces any previous window value. Each segment which passes from SIP to SCM decrements the window. The SIP may request more window space, if the SCM does not automatically update the window.

Interaction Between SIP and Data Link Protocol (Mode VI)

The following are the characteristics of the interface between SIP and the Mode VI (ADCCP) data link protocol (full duplex, binary, synchronous, 32 bit cyclic redundancy code :

- One segment per Mode VI frame.
- A segment always includes a 128 bit BSL followed by zero bits (control segments) up to 4992 bits (data segments).
- Control segments are processed FIFO in both SIP and Mode VI in order to maintain the logic of control segments.
- Data segments may be processed in any order by the SIP, since higher level protocols (TCP and THP) are responsible for segment acknowledgement. However, Mode VI must process them FIFO, since there is an acknowledgement between it and SIP.

SIP - SCM Commands

Commands between SIP and SCM are placed in the BSL. The SIP to SCM commands are the following:

- Data (for user data transmission).
- Echo (for test purposes).
- Request Window (for requesting window space from the SCM, as previously described).
- Subscriber Status (for indicating subscriber operable, going inoperable or busy and subscriber access circuit going inoperable).

The SCM to SIP commands are the following:

- Data (for user data transmission).
- Echo (for test purposes).
- Flow Control - Ready for Next Segment (serves as an acknowledgement to SIP and updates the SIP window).
- Non-Delivery Notice - Undeliverable (destination user down or busy, destination circuit down, or segment discarded by network, as mentioned in the TCP description).

- Error Reject (invalid BSL or command).
- Validation Reject (invalid security, transmission control code, address or precedence).
- SCM Status (for indicating SCM operable, going inoperable or busy and access line operable or going inoperable).

VI. ADVANCED DATA COMMUNICATIONS CONTROL PROCEDURE

The link protocol, the fifth layer protocol from the top, in AUTODIN II is the Advanced Data Communications Control Procedure (ADCCP) [6]. This protocol is responsible for the physical transport of frames on the access channel (SIP to SCM) and packets on the backbone (SCM to SCM). ADCCP is based on the American National Standard for Advanced Data Communications Control Procedures as described in X3S34/589 Draft 6, Revision 2, dated 11 August 1977 [7], and is quite similar to IBM's SDLC. Because of the relationship of ADCCP to the standard and to SDLC, where the terminology "frame" is used, this terminology applies on the access channel. A frame consists of a segment plus beginning and ending flag sequences, address field, control field and frame check sequence. However, once data arrives at an SCM, packets are created and transmitted on the backbone. Packets require additional information to be appended to the segment, including source and destination SCM addresses. It should be noted that ADCCP is synonomous with AUTODIN II Mode VI Protocol.

Mode of Operation

ADCCP uses the balanced asynchronous mode of operation [7]. In this mode, each station of a station pair, called combined stations, can both transmit and receive frames from the other station. Each station of the pair maintains one information transmitting ability to and one information receiving ability from the other station; this is the balanced aspect of the operation. In addition, each station may initiate transmission without receiving permission from the other station;

this is the asynchronous aspect of the operation. Furthermore, ADCCP supports two-way simultaneous operation.

Frame Format

- Flag Sequence:

  Bit series 01111110, signifying the beginning of a frame and the end of a frame. The recognition of data as a flag with this bit sequence is prevented by a bit stuffing technique [7].

- Address Field:

  Address of the destination station for command frames and source station for response frame. A basic address is one octet in length; extended addressing is two octets in length.

- Control Field:

  Contains control information such as commands, responses and sequence numbers. For terrestrial links, a one octet field is used, allowing up to seven unacknowledged frames. For satellite links, where the transmission of an excessive number of acknowledgement frames would result in long delays, a two octet field is used, allowing up to 127 unacknowledged frames.

- Information Field:

  The actual data sent on a link which can vary between zero and 5072 bits on an access channel and 5168 bits on the backbone.

- Frame Check Sequence:

  A 32 bit (32nd degree polynomial), cyclic redundancy code, used as the divisor of transmitted data for error checking purposes. This specification deviates from the 16 bit frame check sequence of the standard.

Commands

- Information (I) Command:

  This is the command for transferring information. It contains the address of the destination station.

- Supervisory (S) Commands:

  There are two S commands -- Receiver Ready (RR) and Receiver Not Ready (RNR). RR is used by a station to indicate it is ready for an I frame and to acknowledge receipt of frames numbered up to and including N(R)-1, where N(R) is the next expected sequence number. RNR is used by a station to indicate that it is busy (cannot accept an I frame) and to acknowledge receipt of frames numbered up to and including N(R)-1. During periods of link inactivity, RR or RNR are sent on the link to ensure that it is operating properly.

- Unnumbered (U) Commands:

  These commands are so called because they do not carry sequence numbers. Two of the commands which are used in AUTODIN II are Set Asynchronous Balanced Mode (SABM) and Set Asynchronous Balanced Mode Extended (SABME). SABM instructs the receiving station to set its send (S) and receive (R) sequence numbers to zero and to clear any error conditions. The optional SABME is the same as SABM, except it uses an extended control field. SABM and SABME commands are acknowledged by using unnumbered acknowledgement (UA) commands (acknowledgement commands which do not carry sequence numbers).

For frame sequence number error recovery purposes, the unnumbered command Reset (RSET) is used. This command sets the Receive Variable (R), whose value is equal to the sequence number expected in the next frame to be received, to zero, clears any error conditions and sets N(R) equal to zero. RSET is acknowledged with a UA response.

Responses

- Supervisory (S) Responses:

  In some instances the N(R) field in the I frame itself is used to acknowledge I frames. However, if the receiving station has no I frames to transmit, acknowledgement cannot occur in this manner. In this case the Receiver Ready (RR) or Receiver Not Ready (RNR) response is used, depending upon whether the receiving station is not busy or busy, respectively. The two responses contain an N(R) field, indicating the sequence number expected in the next frame to be received. This, in effect, acknowledges all received frames with sequence numbers less than N(R). The RR and RNR responses are also used to respond to RR and RNR commands, depending upon whether the receiving station is not busy or busy, respectively.

- Unnumbered (U) Responses:

  These responses are so called because they do not carry sequence numbers. UA responses are used to acknowledge SABM, SABME, and RSET commands.

The Frame Reject Response (FRMR) is used to report error conditions (receipt of an invalid command or response or an I field which exceeds the maximum length) which cannot be recovered from by retransmission.

Acknowledgement and Sequence Numbering System

Each Information (I) frame is sequentially numbered; the maximum is 7 for the basic and 127 for the extended control field format, respectively. These values also correspond to the maximum number of unacknowledged frames at a station; these quantities may be further restricted by frame storage capacities at source and destination stations. Each station maintains a Send Variable (S) on the I frames it transmits and a Receive Variable (R) on the I frames it correctly receives. The value of S is the sequence number of the next I frame to be transmitted; it is incremented by one each time a frame is transmitted. Prior to transmitting an I frame, S is recorded in N(S), the Send Sequence Number; this number is transmitted with I frames. The value of R is the sequence number expected in the next I frame to be received; it is incremented by one when a frame is correctly received and where the value of N(S) in the received frame is equal to R. Prior to transmitting an I or Supervisory (S) frame, R is recorded in N(R), the Receive Sequence Number (sequence number expected on the next I frame to be received); this number is transmitted with I and S frames. The value of N(R) indicates that the transmitting station has correctly received all I frames numbered up to and including N(R)-1.

AUTODIN II acknowledgement of a transmitted frame is not required prior to transmitting the next frame. Rather, many frames can be outstanding (unacknowledged) at a time. For I frames, this number is a maximum of 7 and 127 for the basic and extended control field formats, respectively. Supervisory (S) frame acknowledgements are treated differently. All S frames must be acknowledged by an S response frame Only one S command may be outstanding at a time. An Unnumbered (U)

command is acknowledged by a UA response at the earliest opportunity. Only one U command may be outstanding at a time.

Processing for Exception Conditions and Error Recovery

AUTODIN II link level exception conditions and recovery procedures are as follows:

- Busy Condition:

  This condition occurs when a station can no longer receive I frames. It is signified by transmitting an RNR response with N(R) set to the sequence number of the next frame that is expected to be received. Upon receiving the RNR, the other station will send either an RR or RNR, depending upon whether it is not busy or busy, respectively. This command will continue to be sent until the busy condition clears (an RR response is received).

- Frame Check Sequence (FCS) Error:

  The remainder obtained by dividing the received frame data by the generator polynomial is compared with the Frame Check Sequence (FCS) field transmitted with the frame. If there is inequality, an FCS error has occurred and the frame is discarded.

- Frame Sequence Number Error:

  This error occurs when N(S) in the received frame is not equal to R. After extracting N(R), the frame is discarded.

- Timer Checks:

  When a S or U command is sent, a timer is started. If a response is not received before the timer expires, the command is retransmitted. If three transmissions occur without a response, SIP is notified. Retransmissions occur until SIP directs ADCCP to do otherwise.

If a station has sent the maximum number of unacknowledged I frames, a timer is started. If the timer expires before an acknowledgement is received, an RR command is sent to solicit a response and the timer is restarted. If RR is transmitted three times without obtaining an RR or RNR response, SIP is notified. If a response is obtained within the required time, the acknowledgement information, e.g., N(R) may or may not be correct. If N(R) does acknowledge all outstanding I frames, these frames are purged. If only some of the I frames are acknowledged, these are purged and the remainder are scheduled for retransmission. If the N(R) is out of range, an error situation is assumed; the RSET command is sent and a timer is started. If a UA response is received prior to timer expiration, the outstanding I frames are assigned new sequence numbers, starting with zero, and normal transmission ensues. If RSET is sent three times without receiving a response, SIP is notified of this situation. ADCCP follows this with a similar procedure using the SABM command.

- Frame Reject Conditions:

  A frame may have a correct Frame Check Sequence (FCS) which was discussed previously, but still have invalid data. A Frame Reject Response (FRMR) will be sent under the following conditions:

-- Invalid command or response.

-- Information (I) field which exceeds the maximum length.

-- Invalid frame format.

SIP is notified upon receipt of an FRMR.

- Invalid Receive Sequence Number N(R):

An invalid N(R) occurs when its value is not equal to S, the next send sequence number (e.g., synchronization has been lost between the transmitter and receiver). The detection of this situation causes a RSET command to be sent and a timer started. If the RSET is acknowledged by a UA before the timer expires, any outstanding I frames are assigned new sequence numbers starting with zero and the transmission resumes normally. If the RSET is sent three times without receiving an acknowledgement, SIP is notified. If this procedure fails, ADCCP engages in a similar procedure using the SABM command.

VII. RECOMMENDED SPLICE - AUTODIN II CONNECTION

The following is not intended to be a definitive economic analysis of the proposed SPLICE-AUTODIN II connection versus the conventional connection. Rather, the objective is to highlight the key economic and technical factors. Also, certain economic data, such as the Western Union Telegraph Company rental rate for an MCCU and the line charges for AUTODIN II were not available at the time of writing.

The use of a SPLICE-provided FEP (the standard minicomputer) will avoid the purchase or rental of an MCCU at 21 host sites (number of host sites obtained from [14]). At $21,000 per typical PDP11/34A system [15], the cost avoidance would be approximately $.5 M. In addition, assuming terminals at host sites would be multiplexed or concentrated for connection to the TAC under the conventional AUTODIN II connection, a minimum of 21 line runs from host sites to PSNs would be eliminated by using the recommended connection. Furthermore, in most cases, multiplexed or concentrated terminals at the 23 satellite locations (Multiple Application Processing Systems) [14], would require shorter line runs to the nearest host site than to the nearest PSN.

One of the penalties for making the connection as recommended is the following additional protocol memory allowances which must be provided in the FEP:

- THP :11.104 K bytes
- TCP : 10.016 K bytes
- SIP : 2.48 K bytes
- ADCCP: 4.992 K bytes

28.592 K bytes

Sizes in MCCU under RSX11M Operating System

131 K bytes are available for buffering purposes in the PDP 11/34 MCCU.

The cost of this additional semi-conductor memory for a minicomputer is not large: about $1.5 K or about $32 K for 21 host sites. A much more bothersome aspect is that the protocol software which has been written under DCA contracts is written in two languages--C Programming Language for THP, TCP and SIP, and PDP 11 assembly language for ADCCP. Any user can obtain a copy of all protocol object code from DCA. However, this will only be of benefit to SPLICE if a PDP 11 machine is selected through the SPLICE Procurement process. Likewise, if the selected minicomputer happens to be one for which the C compiler has been implemented, the object code for THP, TCP and SIP could be obtained via compilation.

Yet another possibility is to acquire the software for all protocol layers through the SPLICE Procurement process. A natural choice of language, for a vendor without a C compiler, is PASCAL since it is the chosen system programming language of SPLICE. Rather than specify software development in the hardware acquisition, which would be difficult, or let a separate contract for protocol software, the RFP should specify that the standard minicomputer (FEP) must be capable of providing a single interface to AUTODIN II at each host site for terminal-to-host, terminal-to-terminal, and host-to-host communication. This approach would leave it to the vendor to propose a total minicomputer package which provides the required functions at SPLICE nodes and achieves AUTODIN II compatability as well. The latter might be achieved through direct hardware and object code compatability, by way of a C compiler, or by software development. As a minimum, the highest layers of protocol--TH and HSI--must be provided as part of SPLICE. These protocols are user specific and are not provided with AUTODIN II. Life cycle

software maintenance and cost considerations dictate that software to be developed must be programmed in a single high order language, with no use of assembly language. A corollary to this approach is that the hardware should be sized to have sufficient storage and speed to allow HOL programming rather than resort to the use of assembly language because of hardware constraints.

Recommendation:

The Solicitation Document (SD) should be made more specific with respect to the AUTODIN II interconnect. A mandatory requirement should be incorporated in the SD which calls for the minicomputer FEP to provide a single AUTODIN II interface at each host site which will allow communication between terminals, between hosts, and between terminal and host. Additionally, any protocol software to be provided must be coded in a single HOL acceptable to the Navy.

In order to contrast the conventional AUTODIN II connection and protocol layers with the proposed connection, Figures 5 and 6 depict the two systems, respectively. In the former, the orientation is to terminal-to-host communication. As mentioned previously, in many instances a more general interface is desired -- one that allows and is independent of the types of communication processes, whether they be terminal-to-terminal, host-to-host, or terminal-to-host. A more general design, and one which should be longer lived due to its greater flexibility, is shown in Figure 6. It is suggested that the interface which should have been designed for AUTODIN II should have looked more like Figure 6 than Figure 5.

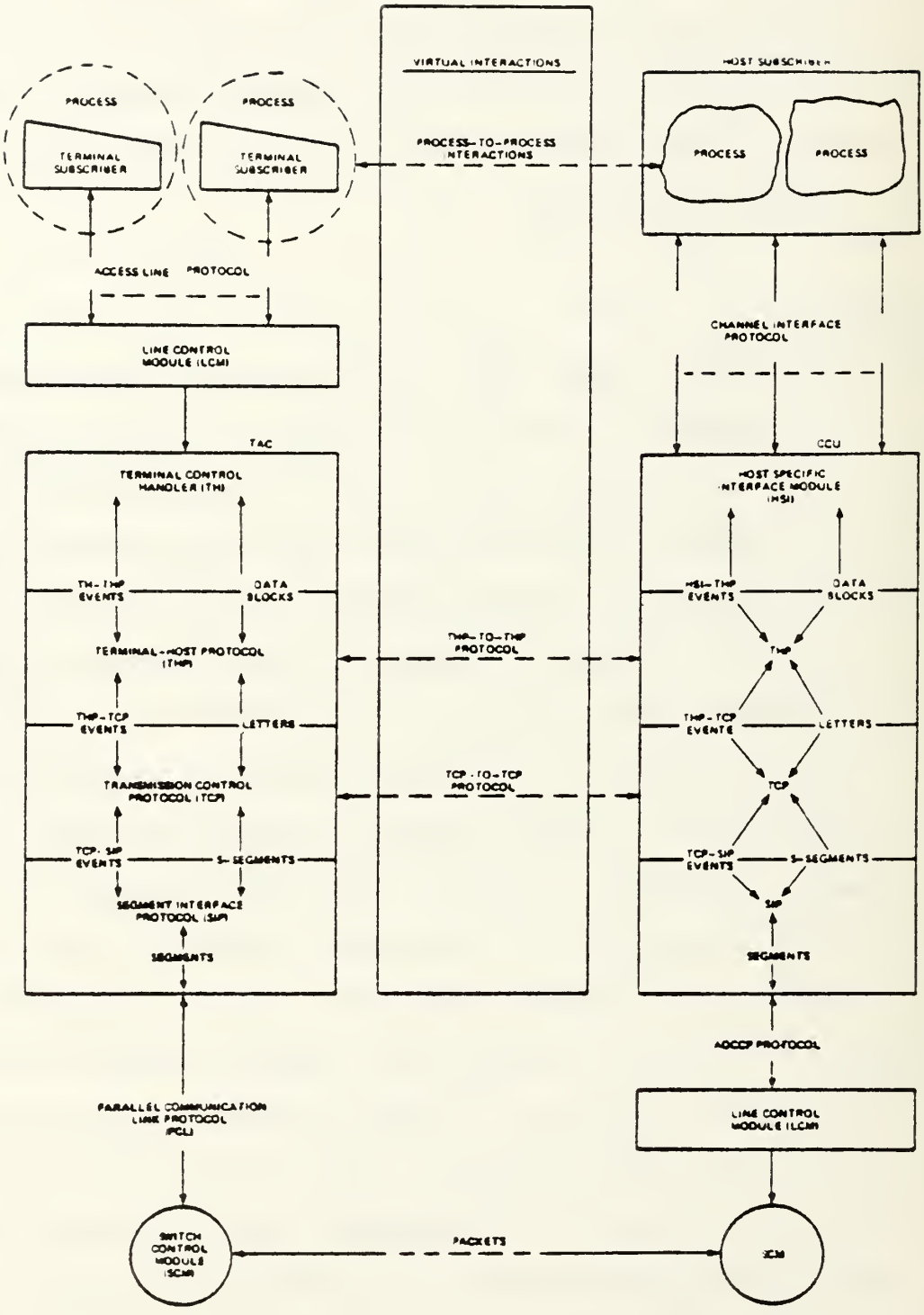
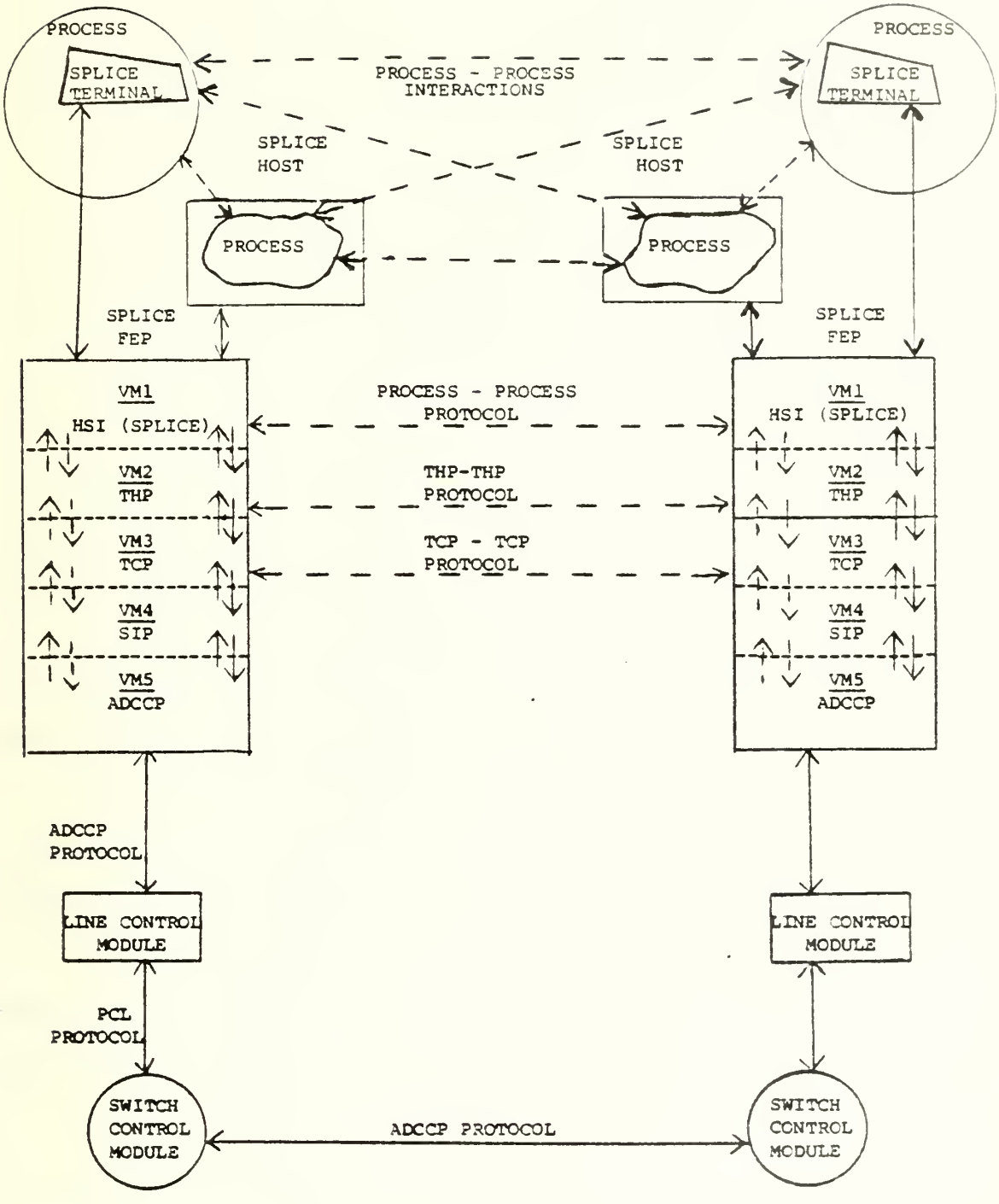


FIGURE 5. Possible Levels of Process-to-Process Protocols
Source: Ref. 8.



Legend:
VM: Virtual Machine
: Software Interrupt
(Implemented on FDX bus)
: Message Transfer
(Implemented on FDX bus)

FIGURE 6. Proposed SPLICE-AUTODIN II Interface

The proposed SPLICE-AUTODIN II Interface, as shown in Figure 6, has the following characteristics:

- All protocol software is implemented in the FEP, which is located at host sites.
- A SPLICE specific protocol layer is defined -- HSI (SPLICE) -- which provides communication at the user process level for terminal-to-host, terminal-to-terminal, and host-to-host communication.
- Each protocol layer is a virtual machine, providing isolation for protocol software development and protection against latent software errors in one layer affecting other layers, software interrupt signalling and message transfer between adjacent layers.
- Protocol interrupt and data transfer between virtual machines is implmented on an FDX bus in the FEP to provide for simultaneous flow of data to and from the network.
- Terminals co-located with hosts and satellite terminals communicate through the FEP with the SPLICE host and with the AUTODIN II network.
- Various modules of the HSI (SPLICE) would be invoked, according to terminal and host requirements at various sites, via parameter entries by the SPLICE Network manager.

Recommendation:

It is recommended that a SPLICE-AUTODIN II interface be acquired with the above specifications and in accordance with Figure 6 via the

SPLICE Procurement process. Note: Procurement Officials may consider the specification of a virtual machine approach too restrictive. If this is the case, "program" could be substituted for "virtual machine" in the SD and a non-virtual machine approach would be utilized.

VIII. NAVY AUTODIN II INTERFACE DESIGN

Overview

A project has been underway for several years to design an interface for connecting Navy systems to AUTODIN II. The specifications and design of this system are documented in references 16 and 17. The major part of the Naval Postgraduate School project effort since the submission of the preliminary report has been spent on evaluating this design and in recommending alternatives, where appropriate.

The primary goal of the design is to implement AUTODIN II protocols on the existing Interdata 7/32 (I 7/32) front-end processors and terminal concentrators. The design was applied to Naval Supply Inventory Control Points (ICP) and the Chief of Naval Personnel Advanced Information System. In the former case, the design was to be provided with the Logistics Data Communication (LDC) system, a predecessor of SPLICE. The discussion in the referenced documents implies that the LDC was operational at the time the documents were written when, in fact, the system was still undergoing development and testing. Discussions with personnel, at FMSO, NAVTASC, NAVTELCOM and DCEC have indicated that this design could serve as the model for interconnecting SPLICE to AUTODIN II, if not the actual hardware and software. The software design is the most comprehensive treatment of interfacing which we have reviewed. Indeed, after evaluating the referenced documents, it was concluded that the software design is useable for SPLICE in part. Unfortunately, the equipment on which the protocols are to be implemented--the I 7/32--is outdated. In addition, the inadequacy of the I 7/32 to handle communication traffic for the ICP Resolicitation Project has been cited [19].

A better hardware system can be obtained through the SPLICE Procurement. The use of a SPLICE Processor will simplify the connection to AUTODIN II at the various supply system nodes, because an additional computer--the I 7/32--will not be required at every connection point. In addition, the microprocessor, which handles the SIP and ADCCP in the I 7/32 design, will also not be required at every connection point [18]. Furthermore, the I 7/32 does not possess virtual memory. This capability would provide a convenient method for logically partitioning the various AUTODIN II protocols, as described in Section VII of this report.

Phase I of the I 7/32 project involved analyzing user requirements and writing functional specifications [16]. In Phase II, software was designed for interfacing Navy systems with AUTODIN II. LDC was a candidate system [16]. It was concluded that the standard MCCU modules: THP, TCP and SIP could be converted to operate on the I 7/32 [16]. In addition, it was concluded that a user specific interface (USI) could be designed to provide functional and software compatibility between user host software and the converted AUTODIN II protocol modules mentioned above [16]. An immediate goal of NAVSUP was to provide communication within the supply community of users, using existing I 7/32 front-end processors [16].

Navy Interface Design Approach

As recommended in Section VII, the interface design eliminates the necessity of installing an MCCU at each SPLICE site and of connecting terminals to the TAC in the nearest PSN, as required by the standard AUTODIN II connection. The major components of the design are shown in Figure 7. It should be noted that the THP, TCP and SIP protocols are

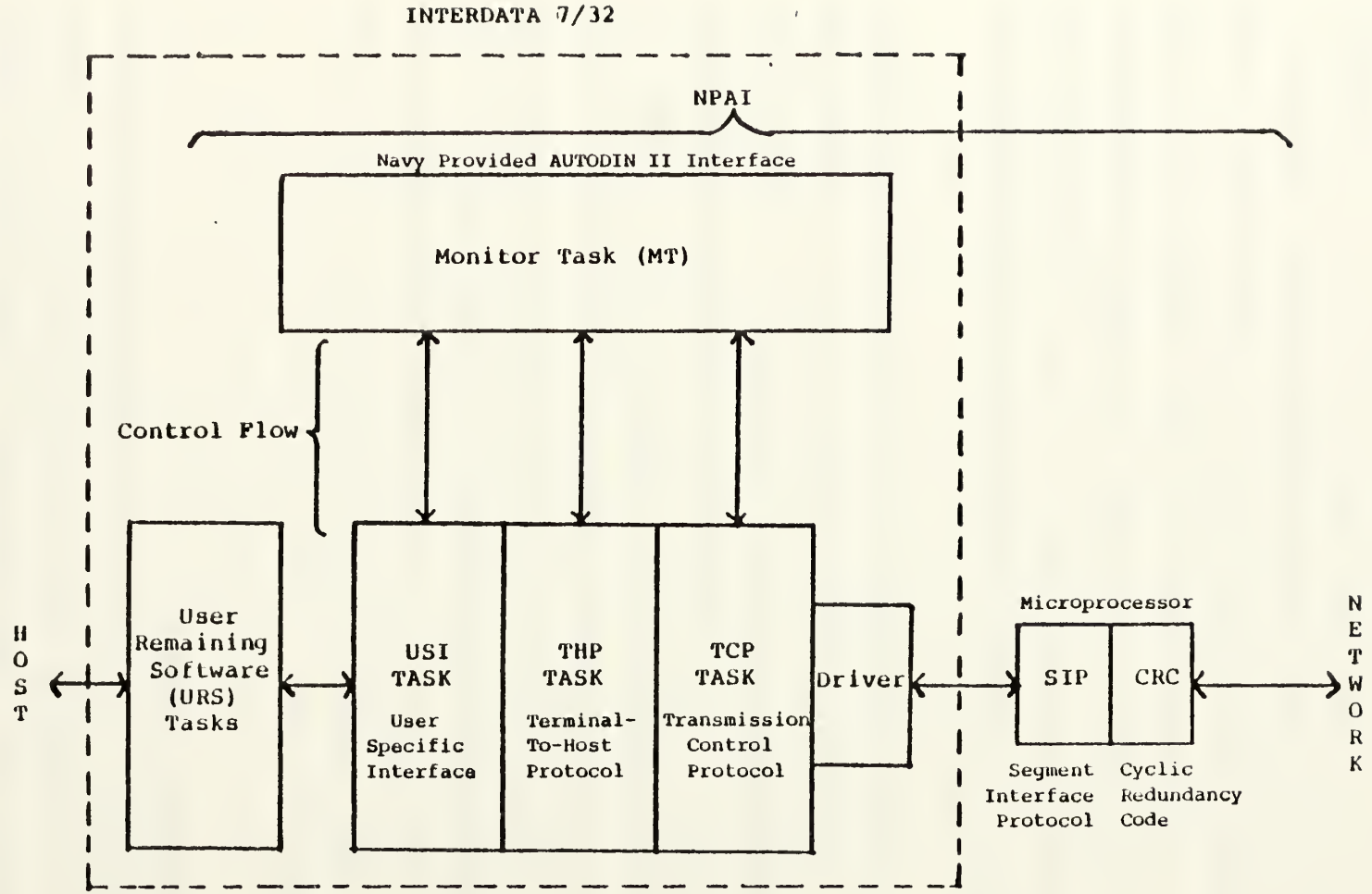


FIGURE 7. Navy Provided AUTODIN II Interface Design

Adapted from Ref. 16.

required along with User Specific Interface, which can also be considered equivalent to the Host Specific Interface as shown in the proposed SPLICE-AUTODIN Interface of Figure 6. This type of protocol arrangement is necessary when terminals are connected to a SPLICE processor and it, in turn, is interfaced to AUTODIN II, as opposed to interfacing terminals directly to AUTODIN II. The HSI (SPLICE) shown in Figure 6 would take the place of the TH and HSI protocols in the standard AUTODIN II connection. Note that HSI (SPLICE) provides for local terminal-to-remote host, local terminal-to-remote terminal, and local host-to-remote host communication. Note that despite the fact that terminals will be connected to a SPLICE Processor in the SPLICE system, as opposed to direct connection to the AUTODIN II Network, the THP or a facsimile must be provided in order to have communication between a local terminal and remote host. Also, the very important Network Virtual Terminal (NVT) component resides in THP. The THP, or its substitute, should be unambiguously specified in the Request for Proposal. Although only the SIP is required for communicating with AUTODIN II and for communication limited to the community of supply users, the SPLICE network will undoubtably evolve into a more comprehensive network, in which case THP and TCP are required. More important, the use of SIP alone would provide a datagram service only; this would be infeasible for file query and update operations. In addition, there would be no accountability and sequence control of packets. The way to fully utilize the capabilities of AUTODIN II is to implement all of its protocols in SPLICE.

The I 7/32 design would expand the 32 virtual connections provided by the standard AUTODIN II MCCU to 96 [16]. However, the I 7/32 would require memory expansion in order to satisfy program and buffer space

requirements [16]. The number of TCP virtual connections which can be supported is also dependent upon the amount of memory available for buffers [16]. Also, in some situations, there may be insufficient memory to support the timer operation [17]. An additional 320 KB of memory is required for each I 7/32 [16]. Other additional equipment includes the microprocessor for SIP and ADCCP and the possible addition of a selector channel at each site, depending upon the number of peripherals currently attached [16]. The selection of the microprocessor (8086 and 8089 units) for the line driver limits the data transfer to 16 bit words. The use of the I 7/32 selector channel bus [17] also limits the data transfer to 16 bit words. A 32 bit data bus would be a plus for handling the AUTODIN II 56 KB/sec. trunk rate.

Concern was expressed regarding possible overloading of the I 7/32 in order to meet AUTODIN II end-to-end message delivery time requirements [16]. Possible solutions which were considered were relocation of application software to another processor; acquisition of an additional I 7/32; and upgrading the I 7/32 to a compatible CPU [16]. With so much concern regarding the adequacy of the I 7/32, it is difficult to understand why it was selected for the AUTODIN II Interface, other than the fact that it was already installed at a number of supply sites and that an open contract existed for acquiring more of these machines. This solution would lock SPLICE into outdated technology for years to come. A better solution would be to acquire a modern computer system via the SPLICE acquisition which would provide the needed capacity over the life of the SPLICE system. Preferably, this standard computer system would offer a virtual storage or virtual machine capability which would alleviate, to an extent, the constraint imposed on the AUTODIN II Interface design by a physical memory size limitation.

As shown in Figure 7, the various protocols execute under the control of the Monitor Task (MT), an Interdata OS/32 operating system. Such an arrangement for SPLICE would complicate software maintenance by introducing yet another operating system into an already complex SPLICE software picture. The procurement of the SPLICE Processor will provide an operating system which can be used for both the AUTODIN II connection and non-AUTODIN II tasks, thus simplifying the software suite.

The software design provides for containing the User Remaining Software (URS) in the I 7/32 (see Figure 7). The URS is never defined. Apparently it is application software. The inclusion of application software in the same processor with the AUTODIN II protocol software should be avoided unless there is a natural partitioning provided in the system, such as the isolation provided by a virtual machine architecture as shown in Figure 6. The reasons for this are the difficulty of maintaining software in a mixed environment, and the competition for available resources and for the attention of the operating system caused by the demands of the applications software, as noted in Reference 16. This unhappy alliance of software types is the result of mixing user oriented software, with its heavy demand for resources, with system software which is dedicated to providing users with quick response on the AUTODIN II Network. Another undesirable mixing of user and system software is the requirement for user software to provide a flow control mechanism involving a maximum number of user-to-network events which may be outstanding [16]. With respect to Figure 7, this function should be provided by the interface USI (User Specific Interface) and not by the URS.

The treatment of AUTODIN II errors seems odd. These are to be handled by Navy hosts and terminals as "dedicated line failures" [16]. The terminology is peculiar because AUTODIN II is a packet switching network which does not provide dedicated lines. Also, there is no mention of the action to be taken in case of failure.

Software Design Aspects

In this section specific aspects of the I 7/32 interface software design are noted, mainly for the purpose of describing deficiencies and improvements which should be corrected and adopted, respectively, if the overall design is to be adopted for SPLICE. As shown in Figure 8, the terminology and software organization are a bit different from that which appears in Figure 7 [17]. User Remaining Software (URS) in Reference 16 has been changed to Existing Navy Application Software (ENAS) in Reference 17. Also, another layer of software--Dispatcher and Message Processing (DSPMP) Subroutine Scheduler--has been inserted in Figure 8 between the Monitor Task (MT) and the protocol programs which appear in Figure 7. There are also some changes relative to the microprocessor line driver function.

One of the problems with the design is that the following task priority relationship is given:

MPSD driver > MT > DSPMP ≥ ENAS [17]

The above implies that DSPMP could have an equal priority with ENAS. In general, system tasks should have a higher priority than application tasks. It should be noted, however, that DSPMP priority can be changed as an operator option.

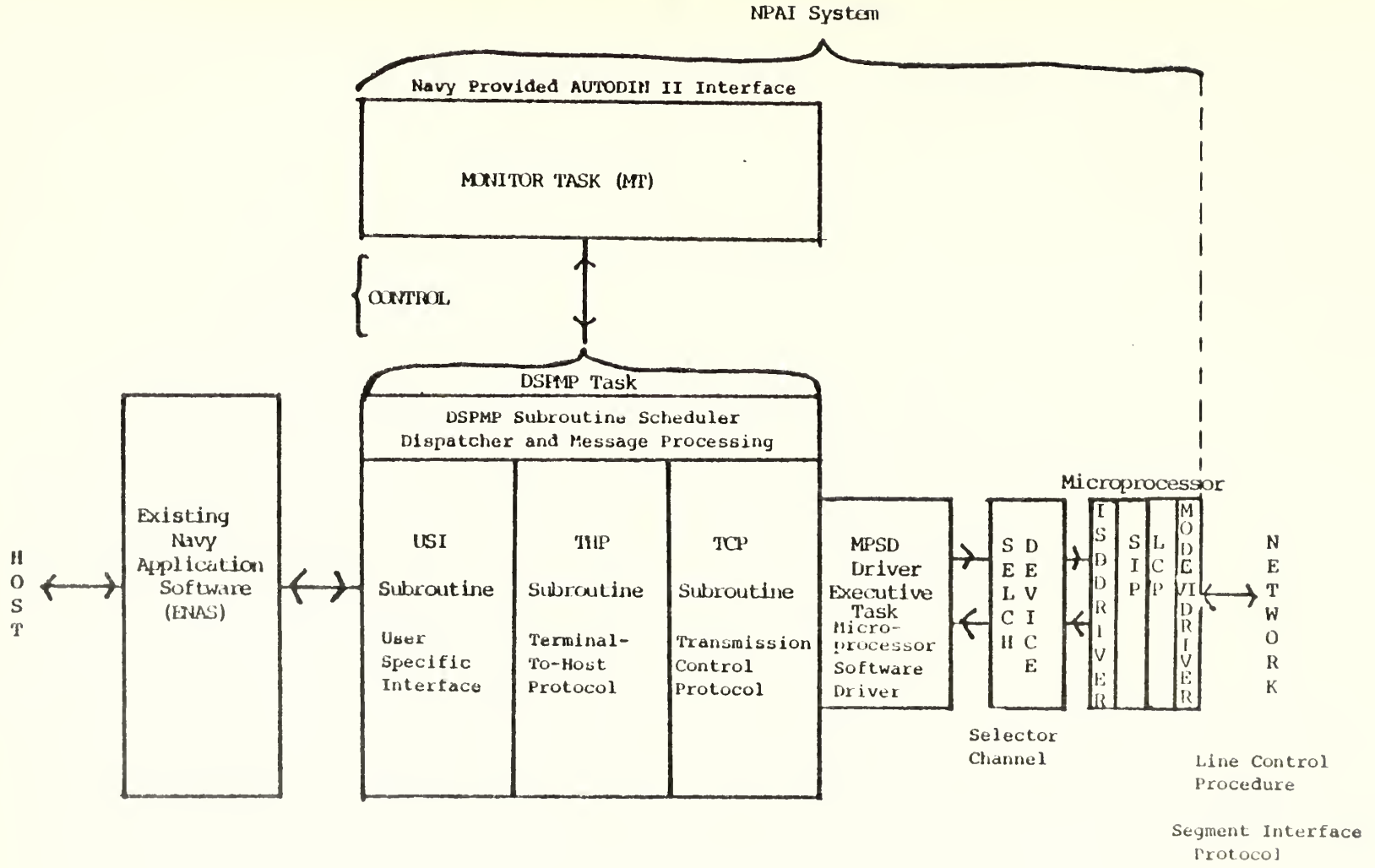


FIGURE 8. Navy Provided AUTODIN II Interface Software Design

Adapted from Ref. 17.

A further problem with the I 7/32 design is that only a 40 KB/sec. throughput is anticipated instead of the AUTODIN II trunk rate of 56 KB/sec. [17]. This is caused in part by the average instruction execution time of 3.5 microseconds, slow by today's standards [17]. This problem could be cured with a modern processor obtained via the SPLICE acquisition. Second, the 40 KB/sec. figure is predicted on the use of the THP binary rather than the NVT mode; the former provides a higher throughput than the latter. Not to use NVT would result in a severe restriction on SPLICE, since only compatible terminal-to-terminal and terminal-to-host communication would be possible. Thus, not only does the Navy interface design provide lower throughput than that capable on AUTODIN II, but it is necessary to achieve this sub-par performance by sacrificing user terminal flexibility.

Proliferation of languages is evident in the use of FORTRAN and the Interdata Common Assembler for development purposes [17]. Although it is stated that only the Interdata Operating System and the protocol software will be needed at operational sites [17], from a software maintenance standpoint two more languages have been introduced. The use of FORTRAN for system programming is also questionable. Recognizing that COBOL is inadequate for system programming, a good choice would be to standardize on PASCAL for all SPLICE system programming efforts, as suggested by FMSO initiatives in this area. More language variety occurs when the needs of developing the microprocessor software are considered. This software will be written in assembly language and a microcomputer high level language, such as PL/M. From a software maintenance standpoint, assembly language should be avoided at all costs; but aside from this argument, there would be two more languages on an

already crowded language scene. Furthermore, the microprocessor driver (MSPD) software will have to be added to the existing I 7/32 driver library at each site, requiring a new system generation at each site. This software must reside in the first 64 KB of the I 7/32 memory [17].

As indicated in a previous section, application software is not always sufficiently isolated from system software in this design. Further evidence of this is indicated by the design of the mechanism for passing messages (events in AUTODIN II terminology) between application software (ENAS) and the AUTODIN II Interface software (NPAI). An NPAI Task Common data area will contain the NPAI list (NPAIL) and the ENAS list (ENASIL) [17]. The former will consist of messages going to ENAS from NPAI and the latter of messages going to NPAI from ENAS [17]. Both NPAI and ENAS are responsible for maintaining their lists and have read access to the other list. To provide better isolation of application software from system software and, hence, improve software maintenance USI, which is a part of NPAI, should maintain the ENASIL and read from NPAIL. The USI would store messages in ENASIL received from ENAS. Other places in the design where USI, rather than ENAS, should perform a function are the following:

- ENAS provides a flow control mechanism in the to-network path by limiting to a fixed value the maximum number of unacknowledged FROM USER events passed to NPAI [17].
- ENAS acknowledges a buffer with data or control information that was passed by NPAI as a TO USER event [17].
- ENAS allocates buffer space for incoming data [17].
- NPAI issues supervisory call via MT to notify ENAS of items to process in the NPAIL.

The above functions are infeasible for implementing in application software at every LDC and SPLICE site because the application software differs. A cleaner interface is provided by putting the above functions in the USI. Naturally, data and control information must ultimately emanate from and be received by the applications software, but the ENAS could receive incoming data from the USI and put data into the USI buffer without specifically being a part of the buffer allocation process for AUTODIN II. In other words, the application software would treat the AUTODIN II connection as another I/O device rather than being considered part of the device. In the I 7/32 design, the ENAS is providing functions which one would expect to find in the USI.

There are certain other aspects of the software design which must be noted because they seem to be errors in documentation or logic, and would have to be corrected if this design is to serve as a model for SPLICE. It is stated that in the Initialization state the Initialization routine invokes other Monitor (MT) routines which always return control back to the Initialization routine [17]. Such a sequence of events--initialization, followed by the beginning of program execution and then an immediate return to initialization--is not the desired sequence nor is this sequence corroborated by the Initialization block diagram [17]. It is also stated that upon receiving an interrupt, the Monitor Trap routine invokes the necessary routine and that one of the possibilities following this action is to wait for the next interrupt [17]. This should only be done if there is no other task waiting to be serviced. Another example is the assumption used in the Line Control Procedure (LCP) (see Figure 8) that the input line buffer space is sufficient to contain the maximum size ADCCP frame coming from the network.

This should be handled by comparing the line buffer size with the size of the incoming frame beforehand and exiting to an error routine, if the frame does not fit.

VIII. NAVY AUTODIN II INTERFACE DESIGN RECOMMENDATIONS

Although it is understood that it is not the intention of NAVSUP to use the Navy Provided AUTODIN II Interface (NPAI) in the SPLICE procurement, much of the software design is useable for the implementation of the AUTODIN II connection on the SPLICE processors. Indeed most of the project effort, since the submission of the preliminary report, has been spent on analyzing the details of the NPAI in order to assess its useability for SPLICE. The extent to which the design can be utilized must await the selection of a SPLICE contractor in order to identify the SPLICE hardware and operating system. It is the hardware of the NPAI which is unsuitable for the following reasons:

- It is unwise to base a new network on outdated hardware technology.
- Inadequate memory size.
- Too much variety of hardware, operating system and programming languages which would lead to difficulties in hardware and software maintenance.
- Less than desired network performance (e.g., 40 KB/sec. vs. 56 KB/sec. desired throughput).
- The I 7/32 hardware constrains SPLICE performance to the characteristics of the OS/32 (MT) operating system.

Although the software design can be used as a model for the SPLICE AUTODIN II connection, the objections to the design raised in the previous section should be addressed. These are the following:

- Achieve greater isolation of application software from system software. Doing this will enhance software maintenance and will improve documentation and make the design easuer to understand.
- Use a single programming language (e.g., PASCAL) for system programming.
- Although the software design is thorough and comprehensive, the documentation is not well structured. It is tedious to read, primarily because it is not structured to reveal the necessary information and does not relegate the details to appendices. The lack of structure in the documentation suggests a lack of top-down design approach in the software itself.
- The task priority scheme should be changed so that operating system tasks always have higher priority than application tasks.

The recommended SPLICE AUTODIN II connection is discussed in Section VI and is shown in Figure 6. As previously indicated, the software design of NPAI could serve as the basis of the SPLICE software design. What this means specifically is that the details of the protocol software (USI, THP, TCP, SIP, ADCCP) described in Reference 17 can be adapted for SPLICE but the hardware (I 7/32 and microprocessor), operating system (OS/32 and MT), and the method of centralized operating system control should not be utilized. The merging of the NPAI software design with the recommended design shown in Figure 6 would have the following characteristics:

- The HSI Of Figure 6 is roughly equivalent to the USI of Figures 7 and 8.

- Each protocol is implemented in a virtual machine (VM) for maximum isolation of protocol layers, which will enhance software development, maintenance and documentation.
- No central operating system control, such as that implemented in NAPI with MT, except for the possibility of a resource manager (e.g., allocation of resources between AUTODIN II and non-AUTODIN II functions in a SPLICE processor ; this level of control would be provided by the SPLICE processor operating system).

A VM would exchange data and control information (e.g., acknowledgements) with another VM without going through a centralized operating system first. In other words, control would be distributed among cooperating peers. This approach would save on operating system overhead. Resources would be reserved, claimed and reservations overridden based on the priority assigned to each VM, i.e., protocol. A higher priority VM would be able to override the previous resource reservation of a lower priority VM and to seize an in-use (claimed) resource including the use of the CPU, when the lower priority VM is finished with the resource. This feature, along with an upper limit on the time slice allocated to each VM (implemented in each VM; each VM sets and manages its own timer) would avoid deadlocks, and if one occurred, it would not be prolonged.

It is beyond the scope of this report to provide all the details regarding a distributed control and operating system design for the SPLICE AUTODIN II connection.

REFERENCES

1. Western Union Telegraph Company, Government Systems Division, AUTODIN II Computer Program Development Specification Terminal-To-Host Protocol (THP) Revision 2, Defense Technical Information Center (ADA081757), 8 Feb. 1980, 296 pages.

2. D.W. Davies, et. al., Computer Networks and Their Protocols, John Wiley and Sons, 1979.

3. Western Union Telegraph Company, Government Systems Division, AUTODIN II Computer Program Development Specification, Transmission Control Program (TCP), Revision 3, Defense Technical Information Center (ADA081756), 8 Feb. 1980, 155 pages.

4. Information Sciences Institute, University of Southern California, DCD Standard Transmission Control Protocol, Defense Advanced Research Projects Agency, Jan. 1980.

5. Western Union Telegraph Company, Government Systems Division, AUTODIN II, Final Computer Program Development Specification: Segment Interface Protocol (SIP), Defense Technical Information Center, (ADA081755), Jan. 1980.

6. Western Union Telegraph Company, Government Systems division, AUTODIN II Mode VI (ADCCP) Line Control Procedures Functional Specification, Defense Technical Information Center, (ADA06747), 2 May 1978, 43 pages.

7. Proposed American National Standard for Advanced Data Communication Control Procedures (ADCCP), Task Group X3S34 on Control Procedures, Technical Committee X3S3 on Data Communications

Committee X3 on Computers and Information Processing, Sixth Draft, Revision 2, 11 August 1977.

8. AUTODIN II Design Executive Summary, Western Union Telegraph Company, Government Systems Division, 18 May 1978, 77 pages.

9. Information for AUTODIN II Subscribers, Western Union Telegraph Company, Government Systems Division.

10. Isadore Lieberman, "AUTODIN II Advanced Telecommunication System," Telecommunications, May 1981, pp. 43-48.

11. AUTODIN II Network Operation Paper, Defense Communications Agency, 14 November 1978, 16 pages.

12. DCA AUTODIN II Briefing, Video Tape Reference Text, NAVTASC Document No. 85C1004, TN-01, Naval Telecommunication Automation Support Center, 12 pages.

13. AUTODIN II, Computer Sciences Corporation, Systems Division, 8 pages.

14. Revised Stock Point Logistics Integrated Communications Environment (SPLICE) Configuration Analysis, Network Analysis Corporation, 28 April 1980.

15. Datapro Reports on Minicomputers, Datapro Research Corporation, June 1981.

16. Interdata 7/32 AUTODIN II Interface, Functional Description, Document No. 50C1002, FD-01, NARDAC, Washington, D.C., August 1979.

17. Interdata 7/32 AUTODIN II Interface, System Specification (Draft) Document No. 50C1002, SS-01, NARDAC, Washington D.C., December 1979.

18. Interdata 7/32 - AUTODIN II Interface Hardware Specifications, Engineering Specifications and Drawings, Document No. 50C1004, ES-01, NAVTASC, Cheltenham, MD, August 1981.

19. Telecommunications Subsystem Project Plan for the Inventory Control Points Resolicitation Project, Naval Supply Systems Command, Washington, DC.

20. Model, 7/32 Processor User's Manual, Perkin-Elmer, Interdata Division, Publication No. 29-405R02, May 1978.

LIST OF ACRONYMNS

ACK	ADKNOWLEDGEMENT
ADCCP	ADVANCED DATA COMMUNICATIONS CONTROL PROCEDURE
AYT?	ARE YOU THERE?
BEL	BELL
BS	BACKSPACE
BSL	BINARY SEGMENT LEADER
CCU	CHANNEL CONTROL UNIT
CR	CARRIAGE RETURN
CRC	CYCLIC REDUNDANCY CODE
DSPMP	DISPATCHER AND MESSAGE PROCESSING
EC	ERASE CHARACTER
EL	ERASE LINE
ENAS	EXISTING NAVY APPLICATION SOFTWARE
ENASIL	EXISTING NAVY APPLICATION SOFTWARE LIST
E-O-L	END OF LINE
FCS	FRAME CHECK SEQUENCE
FEP	FRONT END PROCESSOR
FF	FORMFEED
FIN	FINISH
FRMR	FRAME REJECT RESPONSE
GA	GO AHEAD
HOL	HIGH ORDER LANGUAGE
HSI	HOST SPECIFIC INTERFACE
HT	HORIZONTAL TAB

ICP	INVENTORY CONTROL POINT
ID	IDENTIFICATION
IF	INTERRUPT FUNCTION CHARACTERS
I COMMAND	INFORMATION COMMAND
I FIELD	INFORMATION FIELD
LCM	LINE CONTROL MODULE
LCP	LINE CONTROL PROCEDURE
LDC	LOGISTICS DATA COMMUNICATION SYSTEM
LF	LINEFEED
LTU	LINE TERMINATION UNIT
MCCU	MULTIPLE CHANNEL CONTROL UNIT
MPSD	MICROPROCESSOR SOFTWARE DRIVER
MT	MONITOR TASK
NPAI	NAVY PROVIDED AUTODIN II INTERFACE
NPAIL	NAVY PROVIDED AUTODIN II INTERFACE LIST
N(S)	SEND SEQUENCE NUMBER
NUL	NULL
NVT	NETWORK VIRTUAL TERMINAL
PC	PREFIX CHARACTER
PCL	PARALLEL COMMUNICATION LINK
PSN	PACKET SWITCHING NODE
R	RECEIVE
RCTE	REMOTE ECHO CONTROL COMMAND
RFP	REQUEST FOR PROPOSAL
RNR	RECEIVER NOT READY
RR	RECEIVER READY
RSET	RESET

S	SEND
S COMMAND	SUPERVISORY COMMAND
SABM	SET ASYNCHRONOUS BALANCED MODE
SABME	SET ASYNCHRONOUS BALANCED MODE EXTENDED
SCM	SWITCH CONTROL MODULE
SD	SOLICITATION DOCUMENT
SDLC	SYNCHRONOUS DATA LINK CONTROL
SEQ	SEQUENCE
SI	SHIFT-IN
SIP	SEGMENT INTERFACE PROTOCOL
SN	SEND NOW
SO	SHIFT-OUT
SPLICE	STOCKPOINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT
SYN	SYNCHRONIZATION
T-SEGMENT	TRANSMISSION CONTROL PROTOCOL SEGMENT
TAC	TERMINAL ACCESS CONTROLLER
TCP	TRANSMISSION CONTROL PROTOCOL
TH	TERMINAL HANDLER
THP	TERMINAL TO HOST PROTOCOL
U	UNNUMBERED
UA	UNNUMBERED ACKNOWLEDGEMENT
URS	USER REMAINING SOFTWARE TASKS
USI	USER SPECIFIC INTERFACE
VM	VIRTUAL MACHINE
VT	VERTICAL TAB

DISCLAIMER

The opinions expressed in this report are solely those of the author and do not necessarily represent the views of the Naval Postgraduate School, Department of the Navy, or Department of Defense. Much of this report is based on information gleaned from various AUTODIN II public documents which have been authored by contractor and government organizations. The author does not assume responsibility for the accuracy or validity of information in those documents.

ACKNOWLEDGEMENTS

Acknowledgement is made of the assistance provided on this project by the following individuals (given in alphabetical order):

LCDR Steve Bristow (FMSO)

LCDR Robert Laclede (NAVSUP)

LCDR Clyde Musgrave (DCEC) (Special Thanks)

Ms. Clara Perlingiero (NAVTASC)

Mr. Vic Russell (NAVTELCOM)

Ms. Mary Willoughby (FMSO)

DISTRIBUTION LIST

	No. Copies
Prof. Dushan Badal Code 52Zd Computer Science Department Naval Postgraduate School Monterey, California 93940	1
LCDR Steve Bristow, Comptroller Code 94L Fleet Material Support Office Mechanicsburg, Pennsylvania	1
LT Ted Case Code 94L Fleet Material Support Office Mechanicsburg, Pennsylvania	1
CAPT Chuck Gibfried COMNAVAIRPAC Code 40 Naval Air Station San Diego, California	1
Carl R. Jones, Chairman Code 54Js Administrative Sciences Department Naval Postgraduate School Monterey, California 93940	1
Prof. Miles Kennedy Code 54Kd Naval Postgraduate School Monterey, California 93940	1
LCDR Bob Laclede Commander,Naval Supply Systems Command Code 0415A Washington, D.C. 20379	1
Prof. Norman Lyons Code 54Lb Naval Postgraduate School Monterey, California 93940	1
LCDR Clyde Musgrave Defense Communications Engineering Center Dewey Engineering Building 1860 Wiehle Avenue Reston, Virginia 22090	1

Prof. Norman F. Schneidewind 10
Code 54Ss
Administrative Sciences Department
Naval Postgraduate School
Monterey, California 93940

CDR Bill Walton 1
Aviation Supply Office
700 Robbins Avenue
Philadelphia, Pennsylvania 19111

Prof. Roger Weissinger-Baylon 1
Code 54Ws
Administrative Sciences Department
Naval Postgraduate School
Monterey, California 93940

Ms. Mary Willoughby 1
P.O. Box 94
Mendocino, California 95460

Mr. Gil Young 1
Aviation Supply Office
700 Robbins Avenue
Philadelphia, Pennsylvania 19111

Computer Center Library 1
Code 0141
Naval Postgraduate School
Monterey, California 93940

Computer Science Department 2
Code 52
Naval Postgraduate School
Monterey, California 93940

Defense Technical Information Center 2
Cameron Station
Alexandria, Virginia 22314

Knox Library 4
Code 0142
Naval Postgraduate School
Monterey, California 93940

Office of Research Administration 1
Code 012A
Naval Postgraduate School
Monterey, California 93940

ORAS Library 1
Code 54/55
Naval Postgraduate School
Monterey, California 93940